

A Deep Attention-Based Transformer Framework for Personalized Product Recommendation in E-Commerce Systems

Radhika Adki¹, Ujjwala Salunkhe¹, Neha Jain² and Bhagyashree Lambture¹

¹Department of CSE (Core), School of Computing, MIT Art, Design and Technology University, Pune -412201, India

²Department of Computer Engineering, JSPM's Rajarshi Shahu College of Engineering, Pune, India

Email: adkiradhika@gmail.com

Research Paper

Received: 27 Jan.2026, Revised: 13 Mar. 2026, Accepted: 20 Aug. 2026

Abstract:

Personalized product recommendation systems play a vital role in modern e-commerce platforms by helping users discover relevant products based on their preferences and interaction history. However, traditional recommendation methods such as popularity-based approaches, collaborative filtering, and matrix factorization often struggle to capture complex relationships within user-product interaction data. To address these limitations, this paper proposes an Attention-Based Transformer Model for Personalized Product Recommendation that leverages the self-attention mechanism to effectively learn contextual dependencies among user interactions and product features. In the proposed framework, user and product information are first transformed into embedding representations, which are then processed through multiple Transformer attention layers to identify important interaction patterns and generate accurate recommendations. The proposed model is assessed using widely adopted recommendation metrics, namely Precision@10, Recall@10, NDCG@10, and Hit Rate@10. The experimental findings indicate that the Transformer-based framework consistently surpasses conventional approaches and existing deep learning models, including Popularity-based methods, Collaborative Filtering (CF), Matrix Factorization (MF), and LSTM. In particular, the model attains Precision@10 values of 0.381 for training and 0.362 for validation, along with Recall@10 scores of 0.354 and 0.333. Additionally, it achieves NDCG@10 values of 0.368 (training) and 0.350 (validation), and Hit Rate@10 scores of 0.512 and 0.480, respectively, highlighting its superior capability in accurately identifying and ranking relevant items compared to baseline methods. The results indicate that the attention mechanism effectively captures complex user-product relationships and improves the ranking quality of recommended items. Overall, the proposed Transformer-based recommendation framework enhances personalization capability and provides a scalable solution for next-generation intelligent e-commerce recommendation systems.

Keywords: Personalized Product Recommendation, Transformer Model, Attention Mechanism, User-Product Interaction, Recommender Systems, Top-K Recommendation

1. Introduction

The growth of e-commerce has generated vast user content, including product reviews and ratings [1]. Traditional recommendation systems, such as collaborative filtering [2] and content-based filtering [3], mainly use numerical ratings or item similarity, often ignoring detailed opinions in reviews. Aspect-Based Sentiment Analysis (ABSA) [4] addresses this by identifying product aspects and their sentiment polarity, allowing finer evaluation of features like “battery life” or “design aesthetics.” Mapping aspect-level sentiments to quantitative ratings remains challenging. We propose a hybrid recommendation model combining ABSA with deep learning and graph-based learning. First, review texts are processed using a BiLSTM network with attention [5] to extract aspect-specific sentiment representations, which predict sentiment-based rating scores. Second, these predicted scores are compared with actual user ratings to measure consistency. Third, a Graph Attention Network (GAT) [6] models relationships among products, users, sentiment scores, and ratings, producing embeddings that support recommendations based on both textual sentiment and user-rating similarity. This hybrid approach improves recommendation accuracy and interpretability in dynamic e-commerce environments.

The main contributions of this research can be summarized as follows:

1. We design an ABSA framework using BiLSTM with attention to predict fine-grained sentiment ratings from product reviews.
2. We propose a hybrid GAT that integrates sentiment-derived scores, actual ratings, and product-user interactions for enhanced recommendation accuracy.
3. We perform a comprehensive performance evaluation on benchmark e-commerce datasets to demonstrate the superiority of the proposed approach compared to existing sentiment-based and rating-based methods.

The rest of the paper is organized as follows: Section 2 surveys prior research on recommendation systems, including sentiment-aware and graph-based approaches. Section 3 introduces the proposed attention-based transformer framework along with its mathematical formulation. Section 4 presents the experimental setup, results, and evaluation metrics. Section 5 concludes the paper and discusses possible directions for future research.

2. Related Works

Nguyen et al. [5] proposed a retrieval-based model that utilizes user interaction data to generate relevant product recommendations, emphasizing efficient item retrieval mechanisms. Hallikainen et al. [6] demonstrated that personalized recommendations significantly influence customer purchasing decisions and enhance engagement in online grocery shopping. Zhou and Zou [1] highlighted the strategic role of recommendation algorithms in shaping competition among sellers and guiding consumer purchasing behaviour, while Bastani et al. [7] introduced a learning-based approach that adapts recommendations to evolving user behaviour patterns. Recent advances in deep learning have further improved recommendation accuracy. Zhan and Xu [8] combined sentiment analysis of reviews with product image analysis using parallel neural networks, Vashishth et al. [9] explored AI-enabled content personalization to enhance customer experience, and Karabila et al. [10] leveraged BERT for contextual understanding of user reviews to improve recommendation quality. Researchers have also focused on advanced recommendation architectures. Vullam et al. [11] developed a multi-agent system for collaborative recommendation generation, Hassan et al. [12] examined the moderating role of personalized recommendations in building customer trust, satisfaction, and loyalty, and Singh and Kaunert [13] highlighted AI-based predictive models for hyper-personalized digital marketing. Privacy and security considerations have gained attention as well. Javeed et al. [14] reviewed federated learning-based recommendation systems that preserve user privacy, while Bhardwaj et al. [15] discussed AI-driven personalization strategies for effective customer targeting. Liao et al. [16] proposed a collaborative retail model integrating product relationships and user preferences, and Chiu et al. [17] introduced unsupervised learning-based methods for clustering user preferences to enhance recommendation effectiveness.

2.1 Research Gap

Despite advances in personalized product recommendation systems, several limitations persist. Traditional approaches, such as popularity-based methods, collaborative filtering, and matrix factorization, rely heavily on historical interaction data and often fail to capture complex user-product relationships, suffering from data sparsity, cold-start issues, and limited sequential modelling. Deep learning models like RNNs and LSTMs [18] improve temporal pattern recognition but still struggle with long-range dependencies and contextual relationships in large datasets. Many frameworks focus mainly on structured interaction data, overlooking subtle dependencies among user preferences, browsing patterns, and product attributes. While some systems incorporate sentiment analysis, image features, or multi-agent frameworks, they often increase computational complexity and lack efficient mechanisms to capture global dependencies. As a result, generating highly accurate, context-aware recommendations in large-scale e-commerce environments remains challenging.

2.2 Proposed Solution

To address these limitations, this work proposes an Attention-Based Transformer Model [19] for Personalized Product Recommendation. The framework leverages the Transformer architecture with a self-attention mechanism to capture complex relationships within user-product interaction sequences. Unlike traditional sequential models, the attention mechanism enables the model to focus dynamically on the most relevant interactions, improving the modeling of long-range dependencies and contextual relationships. User interaction data—including browsing history, product views, and purchase behavior—are first transformed into embedding representations that capture latent features of users and products. These embeddings are then processed through multiple Transformer attention layers, which assign adaptive attention weights to different interactions, allowing the model to identify key user preferences and prioritize the most relevant products during recommendation generation.

3. Proposed Attention-Based Transformer Model

The proposed recommendation framework utilizes a transformer-based architecture to model sequential user behaviour and predict personalized product recommendations (Figure 1). The model captures complex dependencies among user interactions using a self-attention mechanism, enabling the system to identify relationships between previously interacted products and potential future purchases.

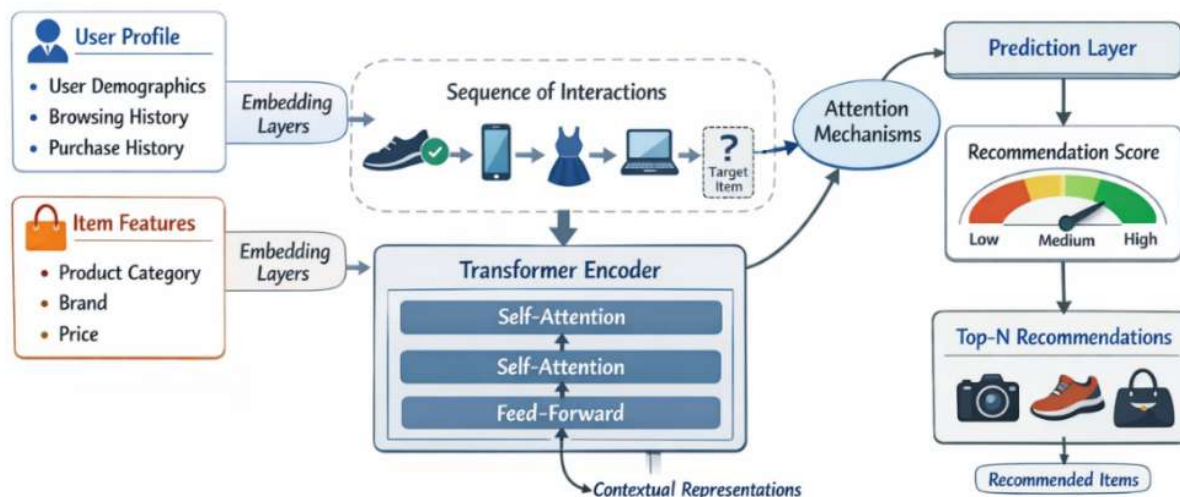


Figure 1: Attention based transformer model for personalized product recommendation

3.1 User Interaction Sequence Representation

In modern e-commerce environments, users interact with online platforms through various activities such as product browsing, purchasing, rating, and reviewing. These interactions generate valuable behavioural data that can be utilized to model user preferences and predict future product interests. A recommendation system aims to analyze these historical interaction patterns to provide personalized suggestions for each user. Let

$$U = \{u_1, u_2, u_3, \dots, u_m\} \quad (1)$$

represent the set of users in the system, where m denotes the total number of users. Similarly, let

$$P = \{p_1, p_2, p_3, \dots, p_n\} \quad (2)$$

represent the set of products available in the recommendation platform, where n denotes the total number of products. Each user interacts with a subset of these products over time, forming a chronological interaction history.

For a given user u , the sequence of interactions can be represented as an ordered list of products based on the time at which the interactions occurred. This ordered sequence can be formally defined as

$$S_u = (p_1, p_2, p_3, \dots, p_T) \quad (3)$$

where, p_i represents the product interacted with at the i^{th} time step, T denotes the total number of interactions recorded for user u , and the ordering reflects the temporal progression of user behaviour.

This sequential representation captures the evolution of user preferences over time. Unlike traditional recommendation models that rely solely on static user-item matrices, sequential modelling allows the system to incorporate contextual information and temporal dependencies between interactions. In practical recommendation scenarios, the interaction sequence may consist of multiple forms of user behaviour such as clicks, purchases, or product views. Therefore, each interaction can also be represented as a tuple containing additional contextual attributes:

$$s_i = (p_i, t_i, a_i) \quad (4)$$

where, p_i denotes the interacted product, t_i represents the timestamp of the interaction, and a_i represents the type of interaction (e.g., click, purchase, or rating). Consequently, the complete interaction sequence for a user can be expressed as

$$S_u = \{(p_1, t_1, a_1), (p_2, t_2, a_2), \dots, (p_T, t_T, a_T)\} \quad (5)$$

This representation allows the recommendation model to incorporate both temporal and behavioural information during the learning process. The primary objective of the recommendation system is to predict the next product that the user is most likely to interact with based on the observed interaction sequence. Formally, the prediction task can be defined as

$$p_{T+1} = f(S_u) \quad (6)$$

where, p_{T+1} represents the predicted next product, and $f(\cdot)$ denotes the recommendation function learned by the model. In this work, the function $f(\cdot)$ is implemented using a transformer-based architecture that learns complex dependencies among products within the interaction sequence. The transformer model processes the sequence S_u to generate a latent representation capturing both short-term and long-term user preferences. Let h_T denote the final hidden representation generated by the transformer encoder after processing the sequence S_u . This representation summarizes the user's historical behavior and serves as the basis for predicting the next product. The probability of recommending a product p_j can therefore be expressed as

$$P(p_j | S_u) = \text{softmax}(h_T W_p + b_p) \quad (7)$$

where, W_p represents the projection weight matrix, b_p is the bias vector, and $\text{softmax}(\cdot)$ converts the output into a probability distribution over all products.

The top-ranked products based on this probability distribution are selected as personalized recommendations for the user. By modelling user interaction sequences in this manner, the system is able to capture dynamic preference patterns and provide more accurate product recommendations. This sequential representation forms the foundation for the proposed attention-based transformer model, which further enhances recommendation performance by learning contextual relationships among products through self-attention mechanisms.

3.2 Product Embedding Layer

In recommendation systems, raw product identifiers are discrete categorical variables that cannot be directly processed by deep learning models. Therefore, it is necessary to convert these symbolic product IDs into continuous vector representations that capture meaningful semantic relationships among products. This transformation is achieved through a product embedding layer, which maps each product into a dense low-dimensional vector space. Let

$$P = \{p_1, p_2, p_3, \dots, p_n\} \quad (8)$$

denote the set of all products in the system, where n represents the total number of products available in the dataset. Each product p_i is represented by a unique index in the product vocabulary. However, such indices do not provide any information about product similarity or user preference patterns. To address this limitation, an embedding layer is introduced to learn distributed vector representations for each product. Formally, the embedding layer can be represented by an embedding matrix

$$E \in \mathbb{R}^{n \times d} \quad (9)$$

where, n represents the number of products, d represents the embedding dimension, and each row of the matrix corresponds to a vector representation of a product. For a given product p_i , the embedding layer maps the product index to a continuous vector representation:

$$e_i = E(p_i) \quad (10)$$

where, E denotes the embedding matrix, $e_i \in \mathbb{R}^d$ represents the embedding vector corresponding to product p_i , and d is the dimensionality of the embedding space.

The embedding vector e_i captures latent semantic properties of the product. During the training process, the parameters of the embedding matrix are optimized such that products with similar interaction patterns or user preferences obtain similar vector representations. Consequently, the embedding layer allows the recommendation model to learn hidden relationships between products that may not be explicitly observable from the raw interaction data.

Given a user interaction sequence

$$S_u = (p_1, p_2, p_3, \dots, p_T) \quad (11)$$

each product in the sequence is transformed into its corresponding embedding vector. This results in a sequence of embeddings defined as

$$X = [e_1, e_2, e_3, \dots, e_T] \quad (12)$$

where, $X \in \mathbb{R}^{T \times d}$ represents the embedding matrix for the entire interaction sequence, T denotes the sequence length, and each row corresponds to the embedding vector of a product in the sequence.

This embedding sequence serves as the input to the transformer-based architecture. By representing products in a continuous vector space, the embedding layer enables the model to capture complex similarity patterns between products. For example, products belonging to the same category or frequently co-purchased by users may be located close to each other in the embedding space. Furthermore, the embedding vectors are learned

jointly with the parameters of the transformer model during training. As the model processes large volumes of interaction data, the embeddings gradually evolve to encode meaningful product relationships such as complementary items, substitute products, or frequently viewed product combinations. Another advantage of the embedding layer is dimensionality reduction. Instead of representing products using high-dimensional sparse vectors (such as one-hot encoding), the embedding layer produces compact dense representations that significantly reduce computational complexity while preserving important information about product relationships.

3.3 Positional Encoding

In sequence-based recommendation systems, the order of user interactions plays a crucial role in understanding user preferences. However, transformer architectures process input elements in parallel and therefore do not inherently preserve sequential order. To address this limitation, positional encoding is incorporated into the input representation to explicitly encode the relative and absolute positions of products within the interaction sequence. Let the embedding sequence obtained from the product embedding layer be represented as

$$X = [e_1, e_2, e_3, \dots, e_T] \quad (13)$$

where T denotes the length of the interaction sequence and $e_i \in \mathbb{R}^d$ represents the embedding vector of product p_i .

To incorporate positional information, a positional encoding vector is added to each embedding vector. The positional encoding is defined using sinusoidal functions:

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{2i/d}}\right) \quad (14)$$

$$PE(pos, 2i + 1) = \cos\left(\frac{pos}{10000^{2i/d}}\right) \quad (15)$$

where, pos represents the position of the product in the interaction sequence, i denotes the embedding dimension index, and d represents the embedding dimension.

The sinusoidal formulation ensures that each position in the sequence obtains a unique encoding while maintaining smooth relationships between nearby positions. This allows the transformer to learn both short-term and long-term dependencies among products. The final input representation for the transformer encoder is obtained by combining the embedding vectors with positional encodings:

$$Z = X + PE \quad (16)$$

where Z represents the position-aware representation of the interaction sequence. By incorporating positional information, the model can effectively distinguish the order of products and capture temporal patterns in user behaviour.

3.4 Self-Attention Mechanism

In the context of product recommendation, self-attention allows the model to identify dependencies among previously interacted products and determine which interactions are most relevant for predicting future preferences. Given the input matrix Z , three different representations are generated for each element in the sequence: query, key, and value vectors.

Query vector is represented as

$$Q = ZW_Q \quad (17)$$

Key vector is represented as

$$K = ZW_K \quad (18)$$

Value vector as

$$V = ZW_V \quad (19)$$

where, W_Q, W_K, W_V are trainable weight matrices.

The query vector encodes the present item's need for contextual insight, whereas the key and value vectors capture information from other elements in the sequence. The relationships among items are then determined through the scaled dot-product attention mechanism, which measures their relative importance.

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (20)$$

where, d_k denotes the dimension of the key vectors. The scaling factor $\sqrt{d_k}$ prevents extremely large dot-product values that could cause instability in the softmax function. The attention mechanism produces a weighted combination of value vectors, where the weights represent the importance of different products in the interaction sequence. As a result, the model can focus on the most relevant past interactions when generating recommendations.

3.5 Multi-Head Attention

Although a single attention operation can capture relationships between sequence elements, it may not be sufficient to model multiple types of product interactions. To overcome this limitation, the Transformer utilizes multi-head attention, enabling it to capture diverse feature representations by attending to multiple subspaces in parallel. For each attention head, separate projection matrices are used:

$$head_i = Attention(QW_Q^i, KW_K^i, VW_V^i) \quad (21)$$

where, W_Q^i, W_K^i, W_V^i represent projection matrices for the i^{th} attention head. Each head independently learns a different type of relationship among products, such as similarity, complementarity, or co-purchase behaviour. The outputs of all attention heads are concatenated and projected to form the final attention output as

$$MultiHead(Q, K, V) = Concat(head_1, head_2, \dots, head_h)W_O \quad (22)$$

where, h represents the number of attention heads, and W_O is a learnable projection matrix. By using multiple attention heads, the model can capture diverse interaction patterns and generate richer representations of user behaviour.

3.6 Feedforward Neural Network

Following the multi-head attention layer, the resulting representation is processed through a position-wise feedforward neural network. This network introduces nonlinearity and enhances the model's ability to learn complex relationships within the sequence representation. The feedforward network is defined as:

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \quad (23)$$

where W_1 and W_2 denote the learnable weight matrices, and b_1 and b_2 represent the corresponding bias vectors. The first linear transformation expands the dimensionality of the input, while the second transformation projects it back to the original dimension. The activation function introduces nonlinear characteristics that enable the model to learn higher-level representations of user preferences.

3.7 Residual Connection and Layer Normalization

Deep neural networks often suffer from training instability due to vanishing or exploding gradients. To mitigate these issues, residual connections and layer normalization are incorporated into the transformer architecture. The residual connection adds the input of a layer to its output as

$$H_1 = \text{LayerNorm}(Z + \text{MultiHead}(Z)) \quad (24)$$

This operation allows gradients to flow directly through the network, improving training stability. Similarly, after the feedforward network, another residual connection is applied:

$$H_2 = \text{LayerNorm}(H_1 + \text{FFN}(H_1)) \quad (25)$$

Layer normalization further stabilizes the training process by normalizing the activations across feature dimensions. These mechanisms enable the transformer model to learn deeper representations effectively.

3.8 Prediction Layer

After passing through multiple transformer encoder layers, the model produces a final hidden representation of the interaction sequence. Let H_T denote the representation corresponding to the last element of the sequence. This representation summarizes the user's historical preferences and is used to predict the probability distribution over all available products.

$$\hat{y} = \text{softmax}(H_T W_p + b_p) \quad (26)$$

where, W_p is the prediction weight matrix, and b_p is the bias vector. The softmax function converts the output scores into a probability distribution across all candidate products.

3.9 Loss Function

To train the recommendation model, the cross-entropy loss function is used. This loss measures the difference between the predicted probability distribution and the actual product that the user interacted with.

$$L = -\sum_{i=1}^N y_i \log(\hat{y}_i) \quad (27)$$

where, y_i represents the ground truth label, and $\hat{y}_i$ represents the predicted probability. The objective of the training process is to minimize this loss by updating the parameters of the embedding layer, attention layers, and feedforward networks using gradient-based optimization algorithms.

3.10 Recommendation Generation

After the training phase, the model can be used to generate personalized product recommendations. For a given user interaction sequence S_u , the trained transformer model produces a probability distribution over all products. The recommendation system ranks the products based on their predicted probabilities and selects the top-N products as recommendations:

$$\text{TopN} = \text{argmax}(\hat{y}) \quad (28)$$

These products represent the items that the model predicts the user is most likely to interact with in the future. The final recommendations are then presented to the user through the e-commerce platform. This approach enables the system to deliver highly personalized product suggestions by effectively modelling sequential user behaviour and capturing complex relationships among products.

4. Simulation Setup

The simulation framework includes dataset preparation, preprocessing steps, model configuration, training parameters, baseline methods, and evaluation metrics. The experiments were designed to assess the capability of the proposed transformer-based architecture in capturing sequential user behaviour and generating accurate product recommendations.

4.1 Dataset Description

The experiments used a large-scale Amazon product interaction dataset [20], containing user-product records of purchases, ratings, browsing history, and review texts. Each record includes a user ID, product ID, product category, rating score, timestamp, and written feedback, providing information on user preferences, product features, and interaction behaviour. Sequential interaction data were created by sorting records chronologically, enabling the model to learn temporal dependencies among product interactions. The dataset's statistical characteristics are summarized in Table 1.

Table 1: Dataset description

Parameter	Value
Number of users	50,000
Number of products	80,000
Total interactions	1,200,000
Average sequence length	24
Maximum sequence length	50

Figure 2, shows the distribution of 80,000 products across ten categories in the Amazon dataset, including Electronics, Books, Clothing, Home, Beauty, Sports, Toys, Automotive, Grocery, and Health. Each category has roughly 8,000 products, and the color-coded bar chart illustrates the fairly balanced distribution, which supports unbiased and effective training of recommendation models.

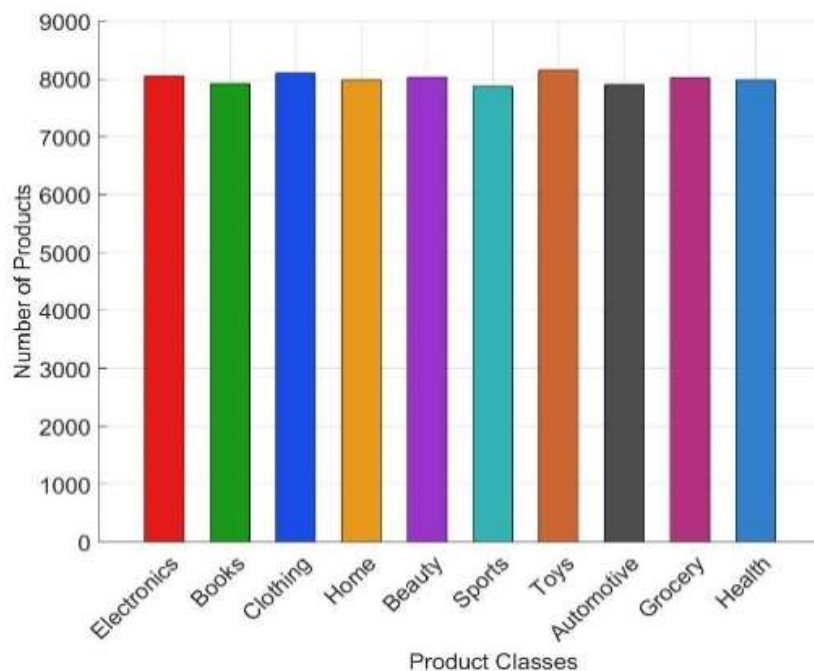


Figure 2: Amazon Product Dataset Class Distribution

4.2 Data Preprocessing

Before training, the dataset was cleaned by removing duplicates and incomplete records. User interactions were chronologically ordered, and sequences were truncated or padded to a fixed length T . Product IDs were converted to numerical indices for embeddings, and the data was split into training (70%), validation (10%), and testing (20%) sets, with each user's last interaction reserved as the target item for evaluation.

4.3 Model Configuration

The Transformer-based recommendation model was developed in Python with the support of deep learning frameworks. Its structure consists of an embedding module, positional encoding, multiple Transformer encoder layers, and an output prediction layer. The main hyperparameter settings are summarized in Table 2.

Table 2: List of hyperparameters

Parameter	Value
Embedding dimension	128
Maximum sequence length	50
Number of transformer layers	2
Number of attention heads	4
Hidden layer dimension	256
Dropout rate	0.2
Batch size	128
Learning rate	0.001

The embedding layer converts product IDs into dense vectors, with positional encoding added to preserve interaction order. Transformer encoder layers capture contextual relationships via multi-head self-attention. Key training parameters are as detailed in Table 3.

Table 3: List of training parameters

Parameter	Value
Optimizer	Adam
Number of training epochs	50
Learning rate	0.001
Loss function	Cross-entropy
Initialization method	Xavier initialization

4.4 Evaluation Metrics

The performance of the recommendation system was evaluated using standard ranking metrics commonly used in recommendation research as shown in Figure 3.



Figure 3: Performance Evaluation Metrics

Figure 4, shows the training and validation loss over epochs for the proposed recommendation model. Training loss decreases steadily, while validation loss follows a similar trend but remains slightly higher due to unseen data. Minor fluctuations arise from stochastic optimization and mini-batch updates. Overall, the smooth decline of both curves indicates stable convergence, effective learning of user-product relationships, and well-regularized training without overfitting.

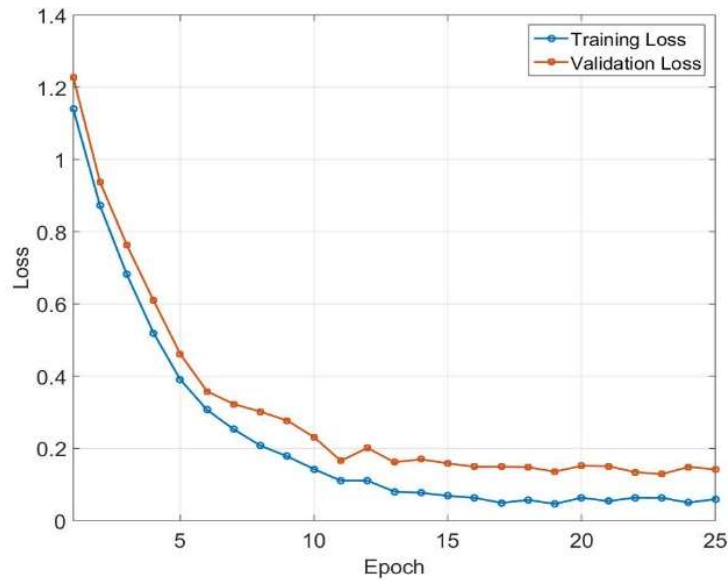


Figure 4: Loss vs. Epochs

Figure 5, shows the attention weight distributions learned by the Transformer-based recommendation model. The heatmaps visualize the importance assigned to different user-product interactions, with darker regions indicating stronger attention. Figure 5(a) depicts the training attention map, where the model gradually emphasizes key interaction positions, capturing relationships between user preferences and product attributes. Figure 5(b) shows the validation map, which maintains a similar structure with slight intensity variations, demonstrating that the model generalizes learned attention patterns to unseen data.

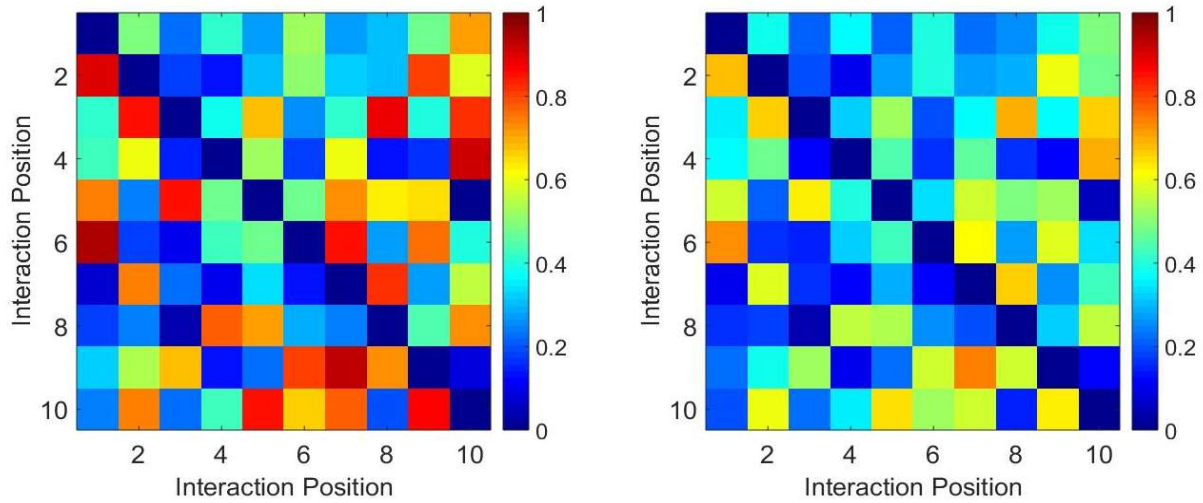


Figure 5: Transformer Attention Map (a) Training (b) Validation.

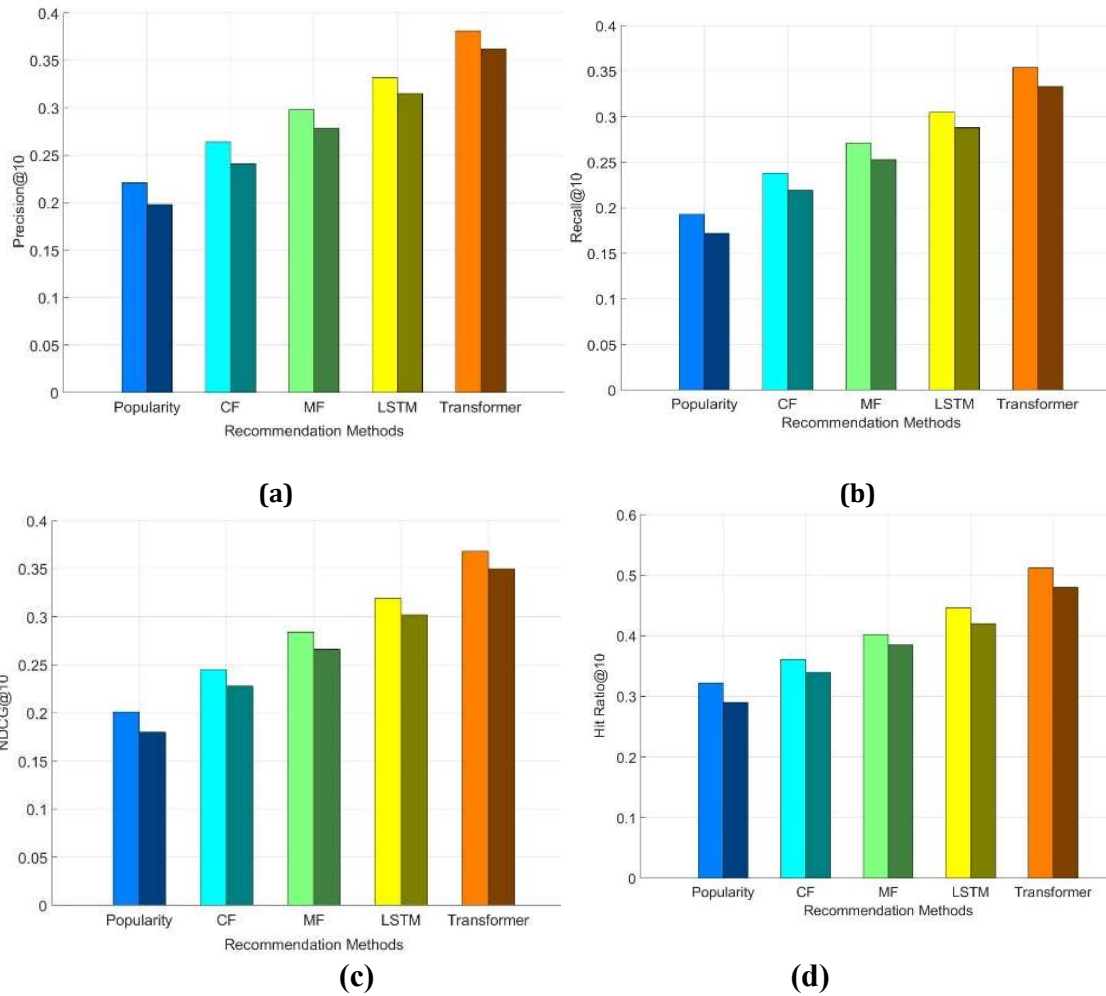


Figure 6: (a) Precision @10 (b) Recall@10 (c) NDCG@10 (d) HitRate@10

Figure 6(a–d) presents a comparative analysis of Precision@10, Recall@10, NDCG@10, and Hit Rate@10 for various recommendation methods across training and validation phases. For Recall@10 (Figure 6b), the popularity-based method shows the lowest performance (0.193 training, 0.172 validation), followed by Collaborative Filtering (0.238, 0.219) and Matrix Factorization (0.271, 0.253). The LSTM-based model improves recall further (0.305, 0.288), while the proposed Transformer-based model achieves the highest performance (0.354, 0.333), indicating superior capability in retrieving relevant items. Similarly, for NDCG@10 (Figure 6c), the popularity-based method records the lowest values (0.201, 0.180), followed by Collaborative Filtering (0.245, 0.228) and Matrix Factorization (0.284, 0.266). The LSTM model attains (0.319, 0.302), whereas the Transformer-based model achieves the highest-ranking quality (0.368, 0.350). For Hit Rate@10 (Figure 6d), the popularity-based method again performs lowest (0.322, 0.290), with improvements seen in Collaborative Filtering (0.361, 0.340) and Matrix Factorization (0.402, 0.385). The LSTM model reaches (0.446, 0.420), while the Transformer-based model achieves the best performance (0.512, 0.480), demonstrating its effectiveness in top-10 recommendations. Overall, the Transformer model consistently outperforms other methods across all metrics with minimal training–validation gaps, indicating strong generalization.

Table 4: Performance Comparison of Recommendation Methods

Method	Precision@10 (Train/Val)	Recall@10 (Train/Val)	NDCG@10 (Train/Val)	HR@10 (Train/Val)
Popularity	0.221 / 0.198	0.193 / 0.172	0.201 / 0.180	0.322 / 0.290
CF	0.264 / 0.241	0.238 / 0.219	0.245 / 0.228	0.361 / 0.340
MF	0.298 / 0.278	0.271 / 0.253	0.284 / 0.266	0.402 / 0.385
LSTM	0.332 / 0.315	0.305 / 0.288	0.319 / 0.302	0.446 / 0.420
Transformer	0.381 / 0.362	0.354 / 0.333	0.368 / 0.350	0.512 / 0.480

Table 4 compares the performance of different recommendation methods. Performance consistently improves from traditional methods (Popularity, CF, MF) to deep learning approaches (LSTM, Transformer). The Transformer-based model achieves the highest scores on both training and validation datasets—Precision@10: 0.381/0.362, Recall@10: 0.354/0.333, NDCG@10: 0.368/0.350, and HR@10: 0.512/0.480—demonstrating its effectiveness in modelling complex user–product interactions and generating accurate personalized recommendations.

5. Conclusion

In conclusion, this paper presents an Attention-Based Transformer Model for Personalized Product Recommendation that effectively addresses the limitations of traditional recommendation methods. By leveraging the self-attention mechanism, the proposed framework captures complex contextual dependencies among user interactions and product features, enabling accurate and personalized recommendations. Experimental results on standard metrics demonstrate that the Transformer-based model significantly outperforms baseline methods, including Popularity, Collaborative Filtering, Matrix Factorization, and LSTM. The improvements highlight the model’s ability to learn meaningful user–product relationships and enhance the ranking quality of recommended items. Overall, the proposed approach provides a scalable and effective solution for intelligent e-commerce recommendation systems, improving user experience and personalization in dynamic online environments.

References

1. Zhou, B.; and Zou, T. (2023). Competing for recommendations: The strategic impact of personalized product recommendations in online marketplaces. *Marketing Science*, 42(2), 360-376.
2. Widayanti, R.; Chakim, M.H.R.; Lukita, C.; Rahardja, U.; and Lutfiani, N. (2023). Improving recommender systems using hybrid techniques of collaborative filtering and content-based filtering. *Journal of Applied Data Sciences*, 4(3), 289-302.

3. Duan, R.; Jiang, C.; and Jain, H.K. (2022). Combining review-based collaborative filtering and matrix factorization: A solution to rating's sparsity problem. *Decision Support Systems*, 156, 113748.
4. Li, C.; Ishak, I.; Ibrahim, H.; Zolkepli, M.; Sidi, F.; and Li, C. (2023). Deep learning-based recommendation system: Systematic review and classification. *IEEE Access*, 11, 113790-113835.
5. Nguyen, D.-N.; Nguyen, V.-H.; Trinh, T.; Ho, T.; and Le, H.-S. (2024). A personalized product recommendation model in e-commerce based on retrieval strategy. *Journal of Open Innovation: Technology, Market, and Complexity*, 10(2), 100303.
6. Hallikainen, H.; Luongo, M.; Dhir, A.; and Laukkanen, T. (2022). Consequences of personalized product recommendations and price promotions in online grocery shopping. *Journal of Retailing and Consumer Services*, 69, 103088.
7. Bastani, H.; Harsha, P.; Perakis, G.; and Singhvi, D. (2022). Learning personalized product recommendations with customer disengagement. *Manufacturing & Service Operations Management*, 24(4), 2010-2028.
8. Zhan, Z.; and Xu, B. (2023). Analyzing review sentiments and product images by parallel deep nets for personalized recommendation. *Information Processing & Management*, 60(1), 103166.
9. Vashishth, T.K.; Sharma, K.K.; Kumar, B.; Chaudhary, S.; and Panwar, R. (2024). Enhancing customer experience through AI-enabled content personalization in e-commerce marketing. *Advances in Digital Marketing in the Era of Artificial Intelligence*, 7-32.
10. Karabila, I.; Darraz, N.; El-Ansari, A.; Alami, N.; and El Mallahi, M. (2024). BERT-enhanced sentiment analysis for personalized e-commerce recommendations. *Multimedia Tools and Applications*, 83(19), 56463-56488.
11. Vullam, N.; Vellela, S.S.; Reddy, V.; Rao, M.V.; and Basha, K.S.K. (2023). Multi-agent personalized recommendation system in e-commerce based on user. In *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, pp. 1194-1199. IEEE.
12. Hassan, N.; Abdelraouf, M.; and El-Shihy, D. (2025). The moderating role of personalized recommendations in the trust-satisfaction-loyalty relationship: An empirical study of AI-driven e-commerce. *Future Business Journal*, 11(1), 66.
13. Singh, B.; and Kaunert, C. (2024). Future of digital marketing: Hyper-personalized customer dynamic experience with AI-based predictive models. In *Revolutionizing the AI-digital landscape*, pp. 189-203. Productivity Press.
14. Javeed, D.; Saeed, M.S.; Kumar, P.; Jolfaei, A.; Islam, S.; and Islam, A.K.M.N. (2023). Federated learning-based personalized recommendation systems: An overview on security and privacy challenges. *IEEE Transactions on Consumer Electronics*, 70(1), 2618-2627.
15. Bhardwaj, S.; Sharma, N.; Goel, M.; Sharma, K.; and Verma, V. (2024). Enhancing customer targeting in e-commerce and digital marketing through AI-driven personalization strategies. *Advances in Digital Marketing in the Era of Artificial Intelligence*, 41-60.
16. Liao, S.-H.; Widowati, R.; and Chan, S.-C. (2025). The retail collaborative recommendations for personalized product recommendations. *International Journal of Retail & Distribution Management*, 53(5), 431-447.
17. Chiu, M.-C.; Huang, J.-H.; Gupta, S.; and Akman, G. (2021). Developing a personalized recommendation system in a smart product service system based on unsupervised learning model. *Computers in Industry*, 128, 103421.
18. Saini, K.; and Singh, A. (2024). A content-based recommender system using stacked LSTM and an attention-based autoencoder. *Measurement: Sensors*, 31, 100975.
19. Xiang, Yafei, Hanyi Yu, Yulu Gong, Shuning Huo, and Mengran Zhu. "Text understanding and generation using transformer models for intelligent e-commerce recommendations." In *Ninth International Symposium on Advances in Electrical, Electronics, and Computer Engineering (ISAEECE 2024)*, vol. 13291, pp. 1292-1297. SPIE, 2024.
20. https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/